

Fragmentos de Código

CUU 1.4 Generar Liquidación

Código página JSP (MenuPrincipal.jsp)

```
<%@page import="entities.Usuario"%>
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<meta http-equiv="content-type" content="text/html; charset=utf-8">
    <meta charset="utf-8">
<title>Menú Principal</title>
    <link href="styles/bootstrap.css" rel="stylesheet">
    <link href="styles/bootstrap.min.css" rel="stylesheet">
    <link href="styles/signin.css" rel="stylesheet">
<%
    Usuario usr = (Usuario)request.getSession().getAttribute("usuario");
    if(usr == null){
        request.setAttribute("titulo", "Acceso Denegado");
        request.setAttribute("mensaje", "Usted no ha iniciado sesión
correctamente o carece de los permisos necesarios apra acceder a esta página.");
        request.setAttribute("pagina", "Login");
        request.setAttribute("direccion", "./index.html");
        request.getRequestDispatcher("/Advertencia.jsp").forward(request,
response);
    }
%>
</head>
<body style="background-color:rgb(251, 252, 255);">
<div class="fixed-top">

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <!-- Brand/logo -->
        <a class="navbar-brand mb-0 h1" href="MenuPrincipal.jsp">Menú</a>

        <!-- Links -->
        <ul class="navbar-nav">
            <li class="nav-item">
                <a class="nav-link" href="ListaPedido.jsp">Pedidos</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="ListaLiquidacion.jsp">Liquidación</a>
            </li>
        </ul>
    </nav>
</div>
</body>
</html>
```

```

        <li class="nav-item">
            <a class="nav-link" href="ListaCliente.jsp">Clientes</a>
        </li>
        <li class="nav-item">
            <a class="nav-link" href="ConsultaAnalisis.jsp">Análisis</a>
        </li>
        <li class="nav-item">
            <a class="nav-link" href="ListaSemilla.jsp">Semillas</a>
        </li>
        <%if(usr.getTipo() == 0){%>
        <li class="nav-item">
            <a class="nav-link" href="ListaUsuario.jsp">Usuarios</a>
        </li>
        <%}%>
    </ul>
</nav>
<br>
<div class="container-fluid">
<div class="jumbotron text-center" >
    <h1>Welcome</h1>
    <br>
    <br>
    <p><b>This system was developed by Pacheco Germán and Angel Folguera</b></p>
</div>
</div>

</div>

</body>
</html>

```

Código página JSP (ListaLiquidación.jsp)

```

<%@page import="entities.Liquidacion"%>
<%@page import="entities.Pedido"%>
<%@page import="java.util.LinkedList"%>
<%@page import="logic.LogicLiquidacion"%>
<%@page import="entities.Usuario"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
    <script
src="https://cdn.jsdelivr.net/npm/jquery@3.5.1/dist/jquery.slim.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js"></script>

<title>Lista Liquidacion</title>

```

```

<!-- Bootstrap core CSS -->
<link href="styles/bootstrap.css" rel="stylesheet">

<!-- Custom styles for this template -->
<link href="styles/signin.css" rel="stylesheet">
<link href="styles/bootstrap.min.css" rel="stylesheet">
<%
    Usuario usr = (Usuario)request.getSession().getAttribute("usuario");
    if(usr == null){
        request.setAttribute("titulo", "Acceso Denegado");
        request.setAttribute("mensaje", "Usted no ha iniciado sesión
correctamente o carece de los permisos necesarios para acceder a esta página.");
        request.setAttribute("pagina", "Menu Principal");
        request.setAttribute("direccion", "./MenuPrincipal.jsp");
        request.getRequestDispatcher("/Advertencia.jsp").forward(request,
response);
    }
    request.getSession().setAttribute("modo", null);
    request.getSession().setAttribute("liquidacion", null);
    LinkedList<Liquidacion> listaLiq= new LogicLiquidacion().getAll();
%>
</head>
<body style="background-color:rgb(251, 252, 255);">
<div class="fixed-top">

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <!-- Brand/logo -->
        <a class="navbar-brand mb-0 h1" href="MenuPrincipal.jsp">Menú</a>

        <!-- Links -->
        <ul class="navbar-nav">
            <li class="nav-item">
                <a class="nav-link" href="ListaPedido.jsp">Pedidos</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="ListaLiquidacion.jsp">Liquidación</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="ListaCliente.jsp">Clientes</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="ConsultaAnalisis.jsp">Análisis</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="ListaSemilla.jsp">Semillas</a>
            </li>
            <%if(usr.getTipo() == 0){%>
            <li class="nav-item">
                <a class="nav-link" href="ListaUsuario.jsp">Usuarios</a>
            </li>
            <%}%>
        </ul>

```

```

</nav>
<br>

<div class="container">
    <div class="mt-4 p-5 bg-info text-white rounded">
        <h1>Lista de Liquidaciones</h1>
    </div>
    <!-- Lista Liquidación -->
    <table class="table table-fixed table-condensed">
        <thead class="table-dark">
            <tr>
                <th>Código Liquidación</th>
                <th>Fecha Liquidación</th>
                <th>Empleado</th>
                <th>Total</th>
                <th>Pedidos</th>
                <th>Editar</th>
                <th>Eliminar</th>
            </tr>
        </thead>
        <tbody>
            <%int index = 0;
            for(Liquidacion l : listaLiq){
            index++;%>
            <tr>
                <td><%=l.getCodLiquidacion()%></td>
                <td><%=l.getFechaLiquidacion()%></td>
                <td><%=l.getEmpleado().getNombreCompleto()%></td>
                <td><%=l.getTotal()%></td>
                <td><button type="button" class="btn btn-info" data-
toggle="collapse" data-target="#demo<%=index%>">Mostrar Pedidos</button></td>
                <td><a class="bg-primary text-white"
href="LiquidacionServlet?accion=editar&codLiquidacion=<%=l.getCodLiquidacion()%>"><butto
n type="button" class="btn btn-primary">Editar</button></a></td>
                <td><a class="bg-danger text-white"
href="LiquidacionServlet?accion=eliminar&codLiquidacion=<%=l.getCodLiquidacion()%>"><but
ton type="button" class="btn btn-danger">Eliminar</button></a></td>
            </tr>
            <thead class="collapse" id="demo<%=index%>">
                <tr>
                    <th></th>
                    <th>Código Pedido</th>
                    <th>Cliente</th>
                    <th>Descuento</th>
                    <th>Fecha Pedido</th>
                    <th>SubTotal</th>
                </tr>
            </thead>
            <tbody class="collapse" id="demo<%=index%>">
                <%for(Pedido p: l.getPedidos()){%>
                <tr>

```

```

        <td></td>
        <td><%=p.getCodPedido() %> </td>
        <td><%=p.getCliente().getRazonSocial()%></td>
        <td><%=p.getDescuento() %></td>
        <td><%=p.getFechaPedido()%></td>
        <td><%=p.GetSubTotal()%></td>
    </tr>
<%} %>
</tbody>
<%} %>
</tbody>
</table>
<button class="btn btn-success" onclick="location.href =
'AgregarLiquidacion.jsp'">Nueva Liquidación</button>
</div>
</div>
</body>
</html>

```

Código servlet (LiquidacionServlet.java)

```

package servlet;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import entities.Liquidacion;
import logic.LogicLiquidacion;

/**
 * Servlet implementation class LiquidacionServlet
 */
@WebServlet("/LiquidacionServlet")
public class LiquidacionServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public LiquidacionServlet() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
    response)

```

```

        */
        protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
            // TODO Auto-generated method stub
            String accion = request.getParameter("accion");
            if(accion!=null)
            {
                switch (accion) {
                    case "eliminar":
                        this.eliminarLiquidacion(request,response);
                        break;
                    case "editar":
                        this.editarLiquidacion(request,response);
                        break;
                    default:
                        break;
                }
            }
        }

        private void eliminarLiquidacion(HttpServletRequest request, HttpServletResponse
        response) throws ServletException, IOException {
            // TODO Auto-generated method stub
            Liquidacion l = new Liquidacion();

            l.setCodLiquidacion(Integer.parseInt(request.getParameter("codLiquidacion")));
            l = new LogicLiquidacion().getByCod(l);
            new LogicLiquidacion().remove(l);
            request.getRequestDispatcher("/ListaLiquidacion.jsp").forward(request,
            response);
        }

        private void editarLiquidacion(HttpServletRequest request, HttpServletResponse
        response) throws ServletException, IOException {
            // TODO Auto-generated method stub
            Liquidacion l = new Liquidacion();

            l.setCodLiquidacion(Integer.parseInt(request.getParameter("codLiquidacion")));
            l = new LogicLiquidacion().getByCod(l);
            request.getSession().setAttribute("liquidacion", l);
            request.getSession().setAttribute("modo", "editar");
            request.getRequestDispatcher("/AgregarLiquidacion.jsp").forward(request,
            response);
        }

        /**
         * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
        response)
         */
        protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
            // TODO Auto-generated method stub

```

```

        doGet(request, response);
    }

}

```

Código página JSP (AgregarLiquidacion.jsp)

```

<%@page import="entities.Estado"%>
<%@page import="entities.Liquidacion"%>
<%@page import="entities.Usuario"%>
<%@page import="java.util.LinkedList"%>
<%@page import="java.util.Date"%>
<%@page import="entities.Pedido"%>
<%@page import="logic.LogicPedido"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Agregar Liquidación</title>
    <!-- Bootstrap core CSS -->
    <link href="styles/bootstrap.css" rel="stylesheet">
    <!-- Custom styles for this template -->
    <link href="styles/signin.css" rel="stylesheet">
    <link href="styles/bootstrap.min.css" rel="stylesheet">
<%
    Usuario usr = (Usuario)request.getSession().getAttribute("usuario");
    if(usr == null){
        request.setAttribute("titulo", "Acceso Denegado");
        request.setAttribute("mensaje", "Usted no ha iniciado sesión
correctamente o carece de los permisos necesarios para acceder a esta página.");
        request.setAttribute("pagina", "Menu Principal");
        request.setAttribute("direccion", "./MenuPrincipal.jsp");
        request.getRequestDispatcher("/Advertencia.jsp").forward(request,
response);
    }

    String modo = (String)request.getSession().getAttribute("modo");
    Liquidacion l = (Liquidacion)request.getSession().getAttribute("liquidacion");
    if(l == null && modo == null){
        l = new Liquidacion();
        l.setPedidos(new LinkedList<Pedido>());
        l.setFechaLiquidacion(new java.sql.Date(new java.util.Date().getTime()));
        request.getSession().setAttribute("liquidacion", l);
    }
    LinkedList<Pedido> pedidosALiquidar = new LogicPedido().getNoLiquidado();
    Usuario u;
    if(modo == null){
        u = (Usuario)request.getSession().getAttribute("usuario");
    }else{
        u = l.getEmpleado();
    }
}

```

```
%>
</head>
<body style="background-color:rgb(251, 252, 255);">
<div class="fixed-top">
  <div class="mt-4 p-5 bg-info text-white rounded text-center">
    <h1>Generar Liquidación</h1>
  </div>
  <div class="container">
    <form action="LiquidacionPedidoServlet" method="post" class="was-validated">
      <div class="form-group">
        <br>
        <h3>Pedidos sin Liquidar</h3>
        <table class="table table-fixed table-condensed">
          <thead class="table-dark">
            <tr>
              <th>Codigo Pedido</th>
              <th>Cliente</th>
              <th>Descuento</th>
              <th>Fecha Pedido</th>
              <th>Acción</th>
            </tr>
          </thead>
          <tbody>
            <%LinkedList<Pedido> listaNo = new LinkedList<Pedido>();
            for(Pedido ped : l.getPedidos()){
              if(ped.getState() != entities.Estado.Deleted){
                listaNo.add(ped);
              }
            }%>
            <%for(Pedido p : pedidosALiquidar ){
              if(!listaNo.contains(p)){%>
                <tr>
                  <td><%=p.getCodPedido() %> </td>
                  <td><%=p.getCliente().getRazonSocial()%></td>
                  <td><%=p.getDescuento() %></td>
                  <td><%=p.getFechaPedido()%></td>
                  <td><a class="bg-primary text-white"
href="LiquidacionPedidoServlet?accion=insertar&codPedido=<%=p.getCodPedido()%>"><butt
on type="button" class="btn btn-primary">Agregar</button></a></td>
                </tr>
              <%}} %>
            </tbody>
          </table>
        </div>
        <div class="form-group">
          <br>
          <h3>Pedidos en la Liquidación</h3>
          <table class="table table-fixed table-condensed">
            <thead class="table-dark">
              <tr>
                <th>Codigo Pedido</th>
                <th>Cliente</th>
```



```

        <th>Descuento</th>
        <th>Fecha Pedido</th>
        <th>Acción</th>
    </tr>
</thead>
<tbody>
<%int i = 0;%>
<%for(Pedido p : l.getPedidos()){
if(p.getState() != Estado.Deleted){%>
    <tr>
        <td><%=p.getCodPedido() %> </td>
        <td><%=p.getCliente().getRazonSocial()%></td>
        <td><%=p.getDescuento() %></td>
        <td><%=p.getFechaPedido()%></td>
        <td><a class="bg-danger text-white"
href="LiquidacionPedidoServlet?accion=eliminar&index=<%=i%>"><button type="button"
class="btn btn-danger">Quitar</button></a></td>
    </tr>
    <%=i++;} %>
</tbody>
</table>
<div>
    <label for="birthday">Total</label>
    <input type="text" id="total" name="total"
value="<%=l.getTotal()%>" readonly>
    <label for="birthday">Empleado Asignado</label>
    <input type="text" id="empleado" name="empleado"
value="<%=u.getNombreCompleto()%>" readonly>
    <label for="birthday">Fecha</label>
    <input type="text" id="date" name="date"
value="<%=l.getFechaLiquidacion()%>" readonly>
</div>
<div>
    <button type="submit" class="btn btn-primary" name="accion"
value="<%= (modo == null ? "insertar_def" : "update_def") %>">Confirmar</button>
    <button type="button" class="btn btn-secondary"
onclick="location.href = 'ListaLiquidacion.jsp'">Cancelar</button>
</div>
</div>
</form>
</div>
</div>
</body>
</html>

```

Código servlet (LiquidacionPedidoServlet.java)

```
package servlet;
```

```
import java.io.IOException;
```

```
import javax.servlet.ServletException;
```

```
import javax.servlet.annotation.WebServlet;
```

```

import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import entities.Pedido;
import entities.Usuario;
import entities.Liquidacion;
import logic.LogicLiquidacion;
import logic.LogicPedido;

/**
 * Servlet implementation class LiquidacionPedidoServlet
 */
@WebServlet("/LiquidacionPedidoServlet")
public class LiquidacionPedidoServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public LiquidacionPedidoServlet() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        // TODO Auto-generated method stub
        String accion = request.getParameter("accion");
        if(accion!=null)
        {
            switch (accion) {
                case "insertar":
                    this.insertarPedido(request,response);
                    break;
                case "eliminar":
                    this.eliminarPedido(request,response);
                    break;
                default:
                    break;
            }
        }
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
response)
     */

```

```

        protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    // TODO Auto-generated method stub
    String accion = request.getParameter("accion");
    if(accion!=null)
    {
        switch (accion) {
            case "insertar_def":
                this.insertar_defLiquidacion(request,response);
                break;
            case "update_def":
                this.update_defLiquidacion(request,response);
                break;
            default:
                break;
        }
    }
    doGet(request, response);
}

    private void update_defLiquidacion(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    Liquidacion l = (Liquidacion)request.getSession().getAttribute("liquidacion");
    new LogicLiquidacion().update(l);
    request.getSession().setAttribute("liquidacion", null);
    request.getRequestDispatcher("/ListaLiquidacion.jsp").forward(request,
response);
}

    private void insertar_defLiquidacion(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    Liquidacion l = (Liquidacion)request.getSession().getAttribute("liquidacion");
    Usuario u = (Usuario)request.getSession().getAttribute("usuario");
    l.setEmpleado(u);
    new LogicLiquidacion().add(l);
    request.getSession().setAttribute("liquidacion", null);
    request.getRequestDispatcher("/ListaLiquidacion.jsp").forward(request,
response);
}

    private void eliminarPedido(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    int index = Integer.parseInt(request.getParameter("index"));
    Liquidacion l = (Liquidacion)request.getSession().getAttribute("liquidacion");
    Pedido p = l.getPedidos().get(index);
    if(p.getState() == entities.Estado.New) {
        l.getPedidos().remove(p);
    }else {
        p.setState(entities.Estado.Deleted);
    }
}

```

```

        }
        request.getRequestDispatcher("/AgregarLiquidacion.jsp").forward(request,
response);
    }

    private void insertarPedido(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        // TODO Auto-generated method stub
        int codPedido = Integer.parseInt(request.getParameter("codPedido"));
        Pedido p = new Pedido();
        p.setCodPedido(codPedido);
        p = new LogicPedido().getByCod(p);
        p.setState(entities.Estado.New);

        ((Liquidacion)request.getSession().getAttribute("liquidacion")).getPedidos().add(p);
        request.getRequestDispatcher("/AgregarLiquidacion.jsp").forward(request,
response);
    }
}

```

Código Lógica (LogicLiquidacion.java)

```

package logic;

import java.util.LinkedList;

import DataBase.DataLiquidacion;
import DataBase.DataPedido;
import entities.Liquidacion;
import entities.Pedido;

public class LogicLiquidacion {
    private DataLiquidacion dl;
    private DataPedido dp;

    public LogicLiquidacion() {
        dl = new DataLiquidacion();
        dp = new DataPedido();
    }

    public LinkedList<Liquidacion> getAll(){
        LinkedList<Liquidacion> listaLiq = dl.getAll();
        for(Liquidacion l : listaLiq) {
            l.setPedidos(dp.getByCodLiquidacion(l));
        }
        return listaLiq;
    }

    public Liquidacion getByCod(Liquidacion liqToSearch) {
        Liquidacion l = dl.getByCod(liqToSearch);
        l.setPedidos(dp.getByCodLiquidacion(l));
        return l;
    }
}

```

```

    }
    public void add(Liquidacion l) {
        dl.add(l);
        for(Pedido p: l.getPedidos()) {
            p.setCodLiquidacion(l.getCodLiquidacion());
            dp.update(p);
        }
    }
    public void update(Liquidacion l) {
        dl.update(l);
        for(Pedido p : l.getPedidos()) {
            switch (p.getState()) {
                case Untouched:
                    break;
                case New:
                    p.setCodLiquidacion(l.getCodLiquidacion());
                    dp.update(p);
                    break;
                case Deleted:
                    p.setCodLiquidacion(0);
                    dp.update(p);
                    break;
                default:
                    break;
            }
        }
    }
    public void remove(Liquidacion l) {
        for(Pedido p : dp.getByCodLiquidacion(l)) {
            p.setCodLiquidacion(0);
            dp.update(p);
        }
        dl.remove(l);
    }
}

```

Código DataBase (DataLiquidacion.java)

```
package DataBase;
```

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.LinkedList;
```

```
import entities.Liquidacion;
import entities.Usuario;
```

```
public class DataLiquidacion {
```

```
    public LinkedList<Liquidacion> getAll(){
        Statement stmt=null;
```

```

        ResultSet rs=null;
        LinkedList<Liquidacion> liquidaciones = new LinkedList<Liquidacion>();
        try
        {
            stmt= DbConnector.getInstancia().getConn().createStatement();
            rs = stmt.executeQuery("select
cod_liquidacion,cod_user,fecha_liquidacion,total from liquidacion");
            if(rs != null)
            {
                while(rs.next())
                {
                    Liquidacion l = new Liquidacion();
                    l.setCodLiquidacion(rs.getInt("cod_liquidacion"));
                    Usuario u = new Usuario();
                    u.setCodUser(rs.getInt("cod_user"));
                    l.setEmpleado(new DataUsuario().getByCod(u));
                    l.setFechaLiquidacion(rs.getDate("fecha_liquidacion"));
                    l.setTotal(rs.getDouble("total"));
                    liquidaciones.add(l);
                }
            }
        }
        catch (SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                if(rs!=null) {rs.close();}
                if(stmt!=null) {stmt.close();}
                DbConnector.getInstancia().releaseConn();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }

        return liquidaciones;
    }

```

```

    public Liquidacion getByCod(Liquidacion liqToSearch) {
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Liquidacion l = null;

        try {
            stmt=DbConnector.getInstancia().getConn().prepareStatement("select
cod_liquidacion,cod_user,fecha_liquidacion,total from liquidacion where cod_liquidacion =
?");

            stmt.setInt(1, liqToSearch.getCodLiquidacion());
            rs=stmt.executeQuery();

            if(rs!=null && rs.next()) {
                l = new Liquidacion();
                l.setCodLiquidacion(rs.getInt("cod_liquidacion"));
            }
        }
    }

```

```

        Usuario u = new Usuario();
        u.setCodUser(rs.getInt("cod_user"));
        l.setEmpleado(new DataUsuario().getByCod(u));
        l.setFechaLiquidacion(rs.getDate("fecha_liquidacion"));
        l.setTotal(rs.getDouble("total"));
    }
} catch(SQLException e) {
    e.printStackTrace();
} finally {
    try {
        if(rs!=null) rs.close();
        if(stmt!=null) stmt.close();
        DbConnector.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
return l;
}

public void add(Liquidacion l) {
    PreparedStatement stmt = null;
    ResultSet keyRs = null;
    try {
        stmt=DbConnector.getInstancia().getConn().prepareStatement("insert
into liquidacion(cod_user, fecha_liquidacion, total) values(?,current_date(),?)",
PreparedStatement.RETURN_GENERATED_KEYS);
        stmt.setInt(1, l.getEmpleado().getCodUser());
        stmt.setDouble(2, l.getTotal());
        stmt.executeUpdate();
        keyRs = stmt.getGeneratedKeys();
        if(keyRs!=null && keyRs.next()) {
            l.setCodLiquidacion(keyRs.getInt(1));
        }
    } catch(SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(keyRs!=null) keyRs.close();
            if(stmt!=null) stmt.close();
            DbConnector.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

public void update(Liquidacion l) {
    PreparedStatement stmt = null;
    try {

```

```

        stmt=DbConnector.getInstancia().getConn().prepareStatement("update liquidacion set
cod_user=?, fecha_liquidacion=?, total=? where cod_liquidacion=?");
        stmt.setInt(1, l.getEmpleado().getCodUser());
        stmt.setDate(2, l.getFechaLiquidacion());
        stmt.setDouble(3, l.getTotal());
        stmt.setInt(4, l.getCodLiquidacion());
        stmt.executeUpdate();
    } catch(SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(stmt!=null) stmt.close();
            DbConnector.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

public void remove(Liquidacion l) {
    PreparedStatement stmt = null;

    try {
        stmt=DbConnector.getInstancia().getConn().prepareStatement("delete
from liquidacion where cod_liquidacion=?");
        stmt.setInt(1, l.getCodLiquidacion());
        stmt.executeUpdate();
    } catch(SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(stmt!=null) stmt.close();
            DbConnector.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
}
}

```

Código DataBase (DataPedido.java)

```

package DataBase;

import java.sql.*;
import java.util.LinkedList;

import entities.Cliente;
import entities.Estado;
import entities.Liquidacion;
import entities.Pedido;
import entities.Semilla;

```



```

public class DataPedido {

    public LinkedList<Pedido> getByCliente(Cliente cli){
        PreparedStatement stmt = null;
        ResultSet rs = null;

        LinkedList<Pedido> pedidos = new LinkedList<>();
        try {
            stmt =
DbConnector.getInstancia().getConn().prepareStatement("select cod_pedido, cuit,
cod_semilla, cod_liquidacion, fecha_pedido, descuento from pedido where cuit=?");
            stmt.setString(1, cli.getCuit());
            rs = stmt.executeQuery();

            if(rs!=null) {
                while(rs.next()) {
                    Pedido p = new Pedido();
                    p.setCodPedido(rs.getInt("cod_pedido"));
                    Cliente c = new Cliente();
                    c.setCuit(rs.getString("cuit"));
                    p.setCliente(new DataCliente().getByCuit(c));
                    Semilla s = new Semilla();
                    s.setCodSemilla(rs.getInt("cod_semilla"));
                    p.setSemilla(new DataSemilla().getByCod(s));
                    p.setCodLiquidacion(rs.getInt("cod_liquidacion"));
                    p.setFechaPedido(rs.getDate("fecha_pedido"));
                    p.setDescuento(rs.getDouble("descuento"));
                    p.setState(Estado.Untouched);

                    pedidos.add(p);
                }
            }
        } catch(SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                if(rs!=null) rs.close();
                if(stmt!=null) stmt.close();
                DbConnector.getInstancia().releaseConn();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        return pedidos;
    }

    public LinkedList<Pedido> getAll(){
        Statement stmt = null;
        ResultSet rs = null;

        LinkedList<Pedido> pedidos = new LinkedList<>();

```

```

        try {
            stmt = DbConnector.getInstancia().getConn().createStatement();
            rs = stmt.executeQuery("select cod_pedido, cuit, cod_semilla,
cod_liquidacion, fecha_pedido, descuento from pedido");

            if(rs!=null) {
                while(rs.next()) {
                    Pedido p = new Pedido();
                    p.setCodPedido(rs.getInt("cod_pedido"));
                    Cliente c = new Cliente();
                    c.setCuit(rs.getString("cuit"));
                    p.setCliente(new DataCliente().getByCuit(c));
                    Semilla s = new Semilla();
                    s.setCodSemilla(rs.getInt("cod_semilla"));
                    p.setSemilla(new DataSemilla().getByCod(s));
                    p.setCodLiquidacion(rs.getInt("cod_liquidacion"));
                    p.setFechaPedido(rs.getDate("fecha_pedido"));
                    p.setDescuento(rs.getDouble("descuento"));
                    p.setState(Estado.Untouched);

                    pedidos.add(p);
                }
            }
        } catch(SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                if(rs!=null) rs.close();
                if(stmt!=null) stmt.close();
                DbConnector.getInstancia().releaseConn();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
    return pedidos;
}

```

```

public LinkedList<Pedido> getByCodLiquidacion(Liquidacion l) {
    PreparedStatement stmt = null;
    ResultSet rs = null;

    LinkedList<Pedido> pedidos = new LinkedList<>();
    try {
        stmt =
DbConnector.getInstancia().getConn().prepareStatement("select cod_pedido, cuit,
cod_semilla, fecha_pedido, descuento from pedido where cod_liquidacion=?");
        stmt.setInt(1, l.getCodLiquidacion());
        rs = stmt.executeQuery();
        if(rs!=null) {
            while(rs.next()) {
                Pedido p = new Pedido();
                p.setCodPedido(rs.getInt("cod_pedido"));

```

```

        p.setListAnalisis(new
DataPedidoAnalisis().getByPedido(p));
        Cliente c = new Cliente();
        c.setCuit(rs.getString("cuit"));
        p.setCliente(new DataCliente().getByCuit(c));
        Semilla s = new Semilla();
        s.setCodSemilla(rs.getInt("cod_semilla"));
        p.setSemilla(new DataSemilla().getByCod(s));
        p.setFechaPedido(rs.getDate("fecha_pedido"));
        p.setDescuento(rs.getDouble("descuento"));
        p.setState(Estado.Untouched);

        pedidos.add(p);
    }
}
} catch(SQLException e) {
    e.printStackTrace();
} finally {
    try {
        if(rs!=null) rs.close();
        if(stmt!=null) stmt.close();
        DbConnector.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
return pedidos;
}

public LinkedList<Pedido> getNoLiquidado(){
    PreparedStatement stmt = null;
    ResultSet rs = null;

    LinkedList<Pedido> pedidos = new LinkedList<>();
    try {
        stmt =
DbConnector.getInstancia().getConn().prepareStatement("select cod_pedido, cuit,
cod_semilla, cod_liquidacion, fecha_pedido, descuento from pedido where cod_liquidacion is
null");
        rs = stmt.executeQuery();
        if(rs!=null) {
            while(rs.next()) {
                Pedido p = new Pedido();
                p.setCodPedido(rs.getInt("cod_pedido"));
                Cliente c = new Cliente();
                c.setCuit(rs.getString("cuit"));
                p.setCliente(new DataCliente().getByCuit(c));
                Semilla s = new Semilla();
                s.setCodSemilla(rs.getInt("cod_semilla"));
                p.setSemilla(new DataSemilla().getByCod(s));
                p.setCodLiquidacion(rs.getInt("cod_liquidacion"));
                p.setFechaPedido(rs.getDate("fecha_pedido"));
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

```

        p.setDescuento(rs.getDouble("descuento"));
        p.setState(Estado.Untouched);

        pedidos.add(p);
    }
}
} catch(SQLException e) {
    e.printStackTrace();
} finally {
    try {
        if(rs!=null) rs.close();
        if(stmt!=null) stmt.close();
        DbConnector.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
return pedidos;
}

public Pedido getByCod(Pedido pedidoToSearch) {
    PreparedStatement stmt = null;
    ResultSet rs = null;
    Pedido p = null;

    try {
        stmt=DbConnector.getInstancia().getConn().prepareStatement("select
cod_pedido, cuit, cod_semilla,cod_liquidacion,fecha_pedido,descuento from pedido where
cod_pedido = ?");

        stmt.setInt(1, pedidoToSearch.getCodPedido());
        rs=stmt.executeQuery();

        if(rs!=null && rs.next()) {
            p = new Pedido();
            p.setCodPedido(rs.getInt("cod_pedido"));
            Cliente c = new Cliente();
            c.setCuit(rs.getString("cuit"));
            p.setCliente(new DataCliente().getByCuit(c));
            Semilla s = new Semilla();
            s.setCodSemilla(rs.getInt("cod_semilla"));
            p.setSemilla(new DataSemilla().getByCod(s));
            p.setCodLiquidacion(rs.getInt("cod_liquidacion"));
            p.setFechaPedido(rs.getDate("fecha_pedido"));
            p.setDescuento(rs.getDouble("descuento"));
            p.setState(Estado.Untouched);
        }
    } catch(SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(rs!=null) rs.close();
            if(stmt!=null) stmt.close();

```

```

        DbConnector.getInstance().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
return p;
}

public void add(Pedido p) {
    PreparedStatement stmt = null;
    ResultSet keyRs = null;

    try {
        stmt=DbConnector.getInstance().getConn().prepareStatement("insert
into pedido(cuit, cod_semilla, fecha_pedido, descuento) values(?,?,?,?)",
PreparedStatement.RETURN_GENERATED_KEYS);
        stmt.setString(1,p.getCliente().getCuit());
        stmt.setInt(2, p.getSemilla().getCodSemilla());
        stmt.setDate(3, p.getFechaPedido());
        stmt.setDouble(4, p.getDescuento());
        stmt.executeUpdate();

        keyRs = stmt.getGeneratedKeys();
        if(keyRs!=null && keyRs.next()) {
            p.setCodPedido(keyRs.getInt(1));
        }
    } catch(SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(keyRs!=null) keyRs.close();
            if(stmt!=null) stmt.close();
            DbConnector.getInstance().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

public void update(Pedido p){
    PreparedStatement stmt = null;
    try {

        stmt=DbConnector.getInstance().getConn().prepareStatement("update pedido set
cuit=?, cod_semilla=?,cod_liquidacion=?,fecha_pedido=?,descuento=? where cod_pedido=?");
        stmt.setString(1, p.getCliente().getCuit());
        stmt.setInt(2, p.getSemilla().getCodSemilla());
        if(p.getCodLiquidacion() == 0) {
            stmt.setNull(3, java.sql.Types.INTEGER);
        }else{
            stmt.setInt(3, p.getCodLiquidacion());
        }
    }
}

```

```

        stmt.setDate(4, p.getFechaPedido());
        stmt.setDouble(5, p.getDescuento());
        stmt.setInt(6, p.getCodPedido());
        stmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(stmt!=null) stmt.close();
            DbConnector.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

public void remove(Pedido p) {
    PreparedStatement stmt = null;
    try {
        stmt=DbConnector.getInstancia().getConn().prepareStatement("delete
from pedido where cod_pedido=?");
        stmt.setInt(1, p.getCodPedido());
        stmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(stmt!=null) stmt.close();
            DbConnector.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
}
}

```

Código Conector (DbConnector.java)

```

package DataBase;

import java.sql.*;

public class DbConnector {

    private static DbConnector instancia;

    private String driver="com.mysql.cj.jdbc.Driver";
    private String host="node93308-env-6563922.jelastic.saveincloud.net";
    private String port="3306";
    private String user="root";
    private String password="LMGhPf8Vhf";
    private String db=" analisis_semillas";
    private int conectados=0;

```

```

private Connection conn=null;

private DbConnector() {
    try {
        Class.forName(driver);
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
}

public static DbConnector getInstancia() {
    if (instancia == null) {
        instancia = new DbConnector();
    }
    return instancia;
}

public Connection getConn() {
    try {
        if(conn==null || conn.isClosed()) {

            conn=DriverManager.getConnection("jdbc:mysql://"+host+":"+port+"/"+db, user,
password);

            conectados=0;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    conectados++;
    return conn;
}

public void releaseConn() {
    conectados--;
    try {
        if (conectados<=0) {
            conn.close();
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

```

Código Entidad (Liquidacion.java)

```

package entities;

import java.sql.Date;
import java.util.LinkedList;

public class Liquidacion {

```

```

private int codLiquidacion;
private Date fechaLiquidacion;
private Double total;
private LinkedList<Pedido> pedidos;
private Usuario empleado;

public Date getFechaLiquidacion() {
    return fechaLiquidacion;
}
public void setFechaLiquidacion(Date fechaLiquidacion) {
    this.fechaLiquidacion = fechaLiquidacion;
}
public Double getTotal() {
    total = (double) 0;
    for(Pedido p : pedidos) {
        if(p.getState() != entities.Estado.Deleted) {
            total += p.GetSubTotal();
        }
    }
    return total;
}
public void setTotal(Double total) {
    this.total = total;
}
public LinkedList<Pedido> getPedidos() {
    return pedidos;
}
public void setPedidos(LinkedList<Pedido> pedidos) {
    this.pedidos = pedidos;
}
public Usuario getEmpleado() {
    return empleado;
}
public void setEmpleado(Usuario empleado) {
    this.empleado = empleado;
}
public int getCodLiquidacion() {
    return codLiquidacion;
}
public void setCodLiquidacion(int codLiquidacion) {
    this.codLiquidacion = codLiquidacion;
}
}

```

Código Entidad (Pedido.java)

```
package entities;
```

```
import java.sql.Date;
```

```
import java.util.LinkedList;
```

```
public class Pedido {
```



```

private int codPedido;
private Date fechaPedido;
private double descuento;
private Cliente cliente;
private Semilla semilla;
private LinkedList<PedidoAnalisis> listAnalisis;
private int codLiquidacion;
private Estado state;

public double GetSubTotal() {
    double total = 0;
    for(PedidoAnalisis pa : listAnalisis){
        if(pa.getState() != entities.Estado.Deleted) {
            total += pa.getAnalisis().getPrecio() ;
        }
    }
    return total*((double)1-(descuento*0.01));
}

public Date getFechaPedido() {
    return fechaPedido;
}
public void setFechaPedido(Date fechaPedido) {
    this.fechaPedido = fechaPedido;
}
public int getCodLiquidacion() {
    return codLiquidacion;
}
public void setCodLiquidacion(int codLiquidacion) {
    this.codLiquidacion = codLiquidacion;
}
public int getCodPedido() {
    return codPedido;
}
public void setCodPedido(int cod_pedido) {
    this.codPedido = cod_pedido;
}
public double getDescuento() {
    return descuento;
}
public void setDescuento(double descuento) {
    this.descuento = descuento;
}
public Cliente getCliente() {
    return cliente;
}
public void setCliente(Cliente cliente) {
    this.cliente = cliente;
}
public Semilla getSemilla() {

```

```

        return semilla;
    }
    public void setSemilla(Semilla semilla) {
        this.semilla = semilla;
    }
    public LinkedList<PedidoAnalisis> getListAnalisis() {
        return listAnalisis;
    }
    public void setListAnalisis(LinkedList<PedidoAnalisis> listAnalisis) {
        this.listAnalisis = listAnalisis;
    }
    public Estado getState() {
        return state;
    }
    public void setState(Estado state) {
        this.state = state;
    }
}

@Override
public boolean equals(Object obj) {
    if (obj == null) {
        return false;
    }

    if (obj.getClass() != this.getClass()) {
        return false;
    }
    Pedido otro = (Pedido) obj;
    if(otro.getCodPedido() != this.getCodPedido()) {
        return false;
    }
    if(this.getState() == entities.Estado.Deleted) {
        return false;
    }
    return true;
}

@Override
public int hashCode() {
    int hash = 5;
    hash = 53 * hash + this.codPedido;
    return hash;
}
}

```

Código Enumeración (Estado.java)

```

package entities;

public enum Estado {
    Untouched, New, Modified, Deleted
}

```